

LIBRARY  
OF THE  
MASSACHUSETTS INSTITUTE  
OF TECHNOLOGY





MODELS FOR PROJECT MANAGEMENT

W. B. Crowston

April, 1970

464-70



MODELS FOR PROJECT MANAGEMENT

W. B. Crowston

April, 1970

464-70





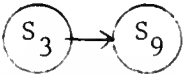
This paper contains a brief survey of the main project management models currently available with a discussion of appropriate solution techniques. The survey begins with the simple time oriented network model that allows the calculation of minimum project length. It is then assumed that each task requires a known amount of a limited resource and the effect of this on project length is examined. Time-cost trade-off models are then introduced. This is followed by an introduction of the usefulness of Markov analysis for the calculation of expected project length in decision networks with probabilistic branches. These models are appropriate in a research or design environment in which the outcomes of a task, and therefore its successive tasks, are not known with certainty. It will be shown that these models are not only useful for decision-making, but also that they are useful in specifying what information should be collected for the decision-maker and in placing a value on that information.

There is another level on which this discussion should be useful. The models that relate job time and cost would allow the efficient centralization of trade-offs that involve these dimensions. In theory, if these were the only relevant dimensions, we could achieve a globally optimal solution. The concept of an explicit trade-off between various dimensions such as time and performance, performance and quality or weight and quality is an idea that might eventually be extended to multi-dimensional global trade-off models. In many instances these pairwise



trade-off models may exist for individual sub-systems perhaps imbedded in a simulation model of that system. What is suggested here is that we bring these models explicitly into the information system and wherever possible include information from the complete project. An example of this might be the implementation of a global weight-cost trade-off model that would show that it is cheaper to buy weight reduction in the landing gear than in the engines, and therefore the weight and budget constraints of both groups should be adjusted. Although we discuss trade-offs and centralization in the time-cost dimension it should be remembered that the basic concepts have more general application.

### The Structure of Project Scheduling Models

The presentation of planning models will be structured according to the types of constraints found in these models. These include the possibility of "time," "resource," and "interdependency" constraints. A time constraint may take the form of an explicit restriction that a task must start on a particular day or that it cannot begin before a given day. More common would be a constraint that the task cannot begin until some other job (a predecessor) is finished. Figure 1 shows twelve tasks,  $S_1 - S_{12}$ , related by constraints of the second kind, that is "precedence" constraints. The graphical notation  implies that  $S_3$  is an immediate predecessor of  $S_9$  and therefore that  $S_9$  could not be started until  $S_3$  is completed [11].

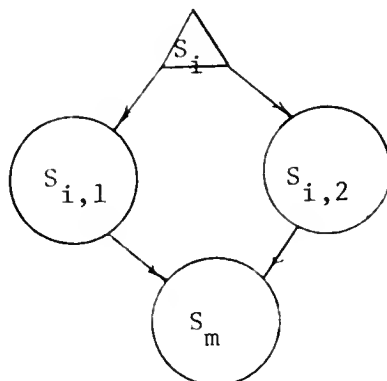


An alternate graphical notation in common use, usually associated with P.E.R.T., techniques [9] represents the tasks as lines rather than nodes and immediate predecessor relations by connection of lines (nodes) rather than by directed lines.



In addition to the precedence relations, the tasks may be related by mutual dependence on a limited resource. If for example in our sample project one individual were required to do both tasks  $S_3$  and  $S_5$ , even though no precedence constraint exists between them, he would be forced to do them serially. He could perform  $S_3$  and then  $S_5$  or  $S_5$  and then  $S_3$  but not both of them together. For problems of practical interest the number of feasible sequences is large and the problem of finding a "best" sequence is a difficult combinatorial problem.

Finally the tasks may be related by the technical nature of the project. It is conceivable that there are several methods of performing some task. This could involve the use of different materials, different production processes and perhaps the performance of the task on regular time "in-house," or on a sub-contract basis. We will indicate this mutually exclusive type of interdependency graphically with a decision node



where  $S_{i,1}$  and  $S_{i,2}$  are two methods of performing task  $S_i$



BASIC C.P.M. GRAPH

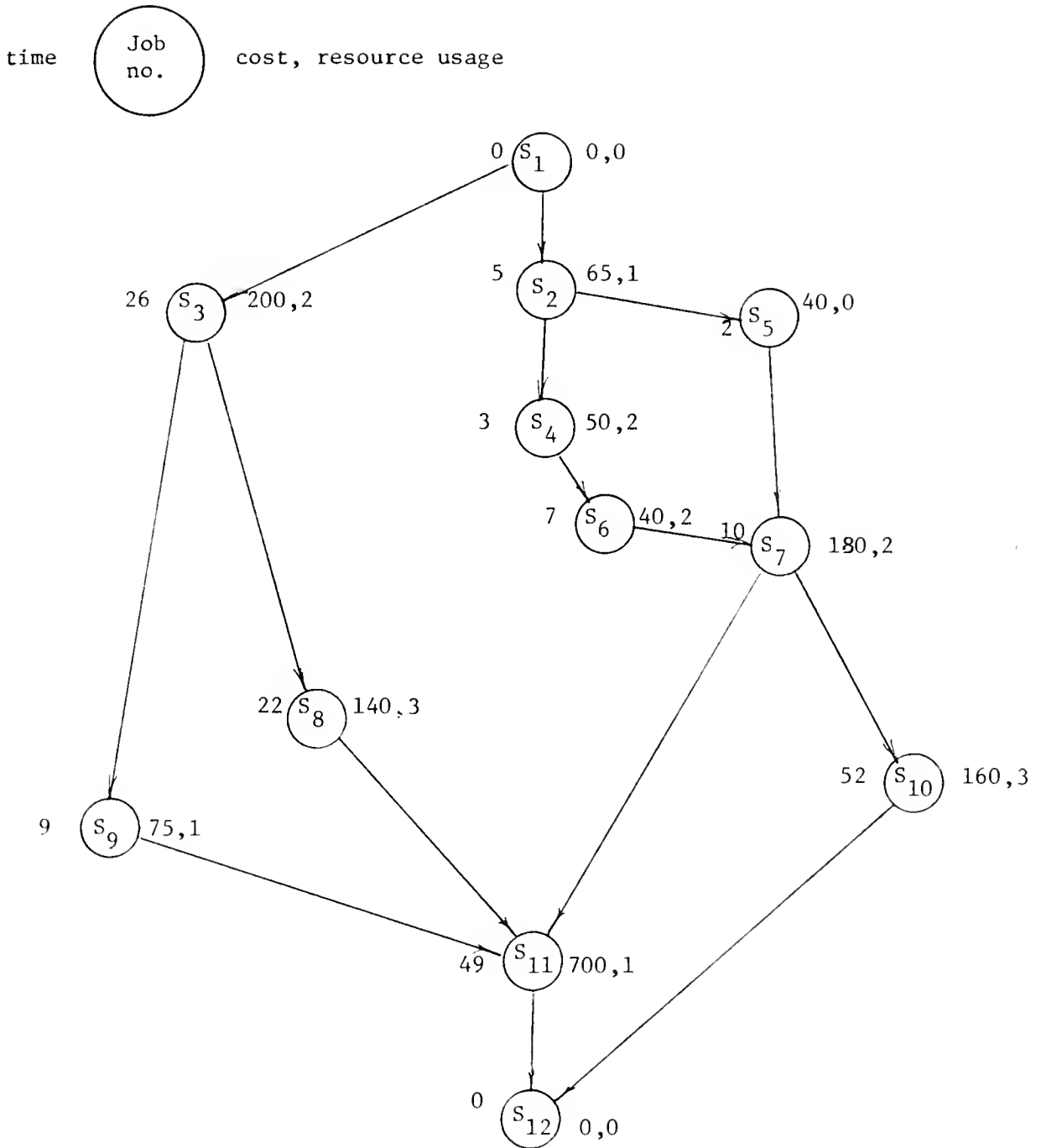
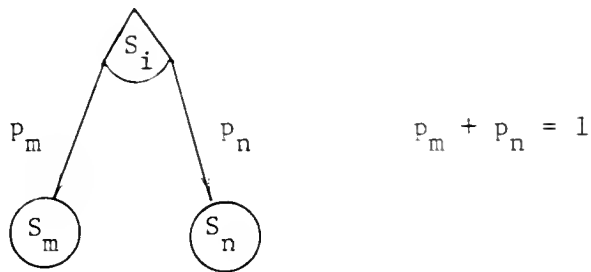


FIGURE 1





Also we may wish to add interdependency constraints that relate decision jobs at different decision nodes. For example, if we subcontract task  $S_i$ , then we should also subcontract job  $S_q$ . Finally we will illustrate our graphical notation for a probabilistic decision node. To show that job  $S_i$  may be followed by either  $S_m$  or  $S_n$  with probabilities  $p_m, p_n$ , we show



We will now briefly present project planning models that are composed of various combinations of these constraints.

### Simple Time Constraints

The basic PERT [9] or CPM [11] model is simply a set of tasks,  $S_i$ , related by precedence constraints.

If we define

$w_i$  = start time of task  $S_i$

$t_i$  = duration of task  $S_i$

then it is possible to show the precedence relationship



as

$$w_i + t_i \leq w_m$$



The minimum length of the project in Figure 1 may be now determined by the following linear programming formulation, with one constraint for each immediate predecessor relationship.

Minimize  $w_{12}$

subject to

$$\begin{array}{ll} \text{Time Constraints} & w_1 + t_1 \leq w_2 \\ & w_1 + t_1 \leq w_3 \\ & w_2 + t_2 \leq w_4 \\ & \quad " \quad " \\ & \quad " \quad " \\ & w_{11} + t_{11} \leq w_{12} \\ & \\ & w_i \geq 0 \end{array}$$

We do not suggest this method for calculating the minimum project length (critical path length) since the usual longest path algorithm [11] is more efficient and also provides information on "earliest" and "latest" start times for each task in the network. Here we define "early" start time,  $E.S._i$ , the earliest time task  $S_i$  can begin if all immediate predecessors of  $S_i$  start at their early start time. Late start,  $L.S._i$ , is defined as the last day on which job  $S_i$  can begin and still not delay the finish of the project beyond its due date, D.D. In Figure 2 we show  $ES_i$ ,  $LS_i$ , and  $slack = LS_i - ES_i$  for all tasks in our sample problem, given an early start for the first job of day 1, and a due date,  $DD = 98$ .



E.S. - L.S. CALCULATION

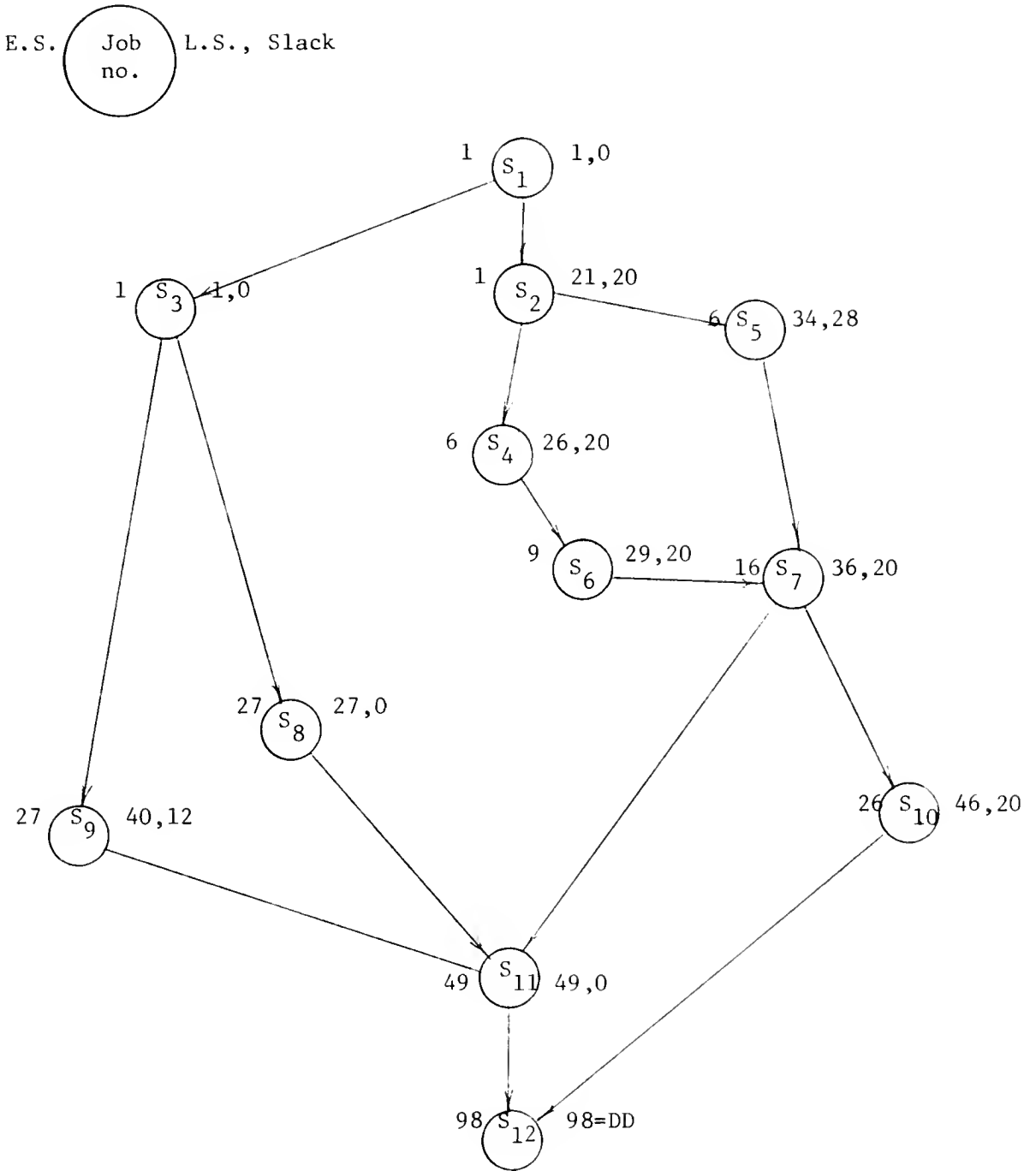


FIGURE 2



## Models with Time and Resource Constraints

Many interesting sequencing problems may be described by time and resource constraints. Among these are models of interest in project scheduling, job-shop scheduling and assembly-line balancing problems. Each of these has been formulated as an integer linear programming problem, but because of the high number of constraints involved and the size of practical problems, it is not recommended as a solution technique. For example, Wiest [16] estimates that a project with 55 jobs using 4 resources over a time span of 30 days would have 5,275 equations and 1,650 variables. The essential problem is that the level of resource is constrained by period and the tasks, given the usual time constraints, may shift through time. Thus in a programming formulation it is necessary to include the possibility that each job could begin on many days and to write constraints that prohibit the total amount of each resource used in any day to be lower than or equal to the resource limit. As a result of the difficulty of this combinatorial problem, the problem is solved in practice by heuristic methods. We will discuss briefly three of these.

The method of Levy, Thompson and Wiest [12] concentrates on reducing the peak usage of the project resource. The jobs from all projects to be simultaneously scheduled are scheduled at Early Start and then the daily demand for each resource is plotted. For the first resource an initial trigger level is set, one unit below the maximum usage. It is assumed that the resources are ordered based on some priority system, perhaps daily





cost per unit. Then all the jobs contributing to the particular peak and in addition having enough slack so that they may be scheduled beyond the peak are listed. Then one of the jobs is chosen probabilistically, the probabilistic weight being proportional to the job's slack time, and the job is shifted a random amount within the slack, forward. The resource peak again is calculated and a lower trigger level is set. This process continues for the first resource until no further improvement is realized. Then freezing the lowest feasible limit for the resource, the procedure is repeated for the second resource and so on through all resources. Now the jobs from each project are segregated and again an attempt is made to shift them and reduce the aggregate trigger level for all projects. Finally, when no further improvement is possible, the final schedule and trigger levels are stored and the process is completely repeated. Due to the probabilistic element in the decision rule, a new schedule will result. After several repetitions, the best schedule of those generated can be chosen.

The next heuristic to be discussed will be scheduling to meet stated resource constraints. The early solutions to problems of this type were obtained with Gantt charts and their use in job-shop scheduling continues. The resource constraint in the job-shop problem will be the limit on machine availability, and the time constraints are provided by technological ordering of operations on a particular work order.

Several heuristic approaches have been suggested for solving the fixed resource problem in the project scheduling context. None of the approaches guarantees an optimal solution, nor consistently outperforms the others. Representative of one approach is an article by Kelley [17]. The routine may be summarized as follows:



1. Technologically order the jobs, calculate early start, late start, and slack. Within the technological ordering, reorder by slack, lowest slack first.

2. Start with the first activity and continue down the list.

- (a) Find the early start of the activity.

- (b) Schedule the activity at early start if sufficient resources are available. If resources are not available, two alternative routines are suggested, c1, c2.

- (c1)Serial Method: Begin the job at the earliest time that resources are available to work it for one day. The job may be split if necessary.

- (c2)Parallel Method: Find the set of all jobs causing the resource violation and rank them by total slack. Delay a sufficient number of jobs with the highest slack so that those remaining on the list can be scheduled.

3. Repeat step 2 for all jobs.

4. Repeat the whole routine with various orderings of the technological list.

In addition to the possibility of allowing job-splits, Kelley suggests that we also allow resource limits to be violated by small amounts. The basic heuristic in the original job ordering and in the parallel loading technique is the use of job slack as a criterion for shifting jobs forward. This is a reasonable approach similar to that used by Levy [12].

A second approach, detailed in Moder and Philips [18], uses Maximum Remaining Path Length, or equivalently Late Start, as a basis for ordering



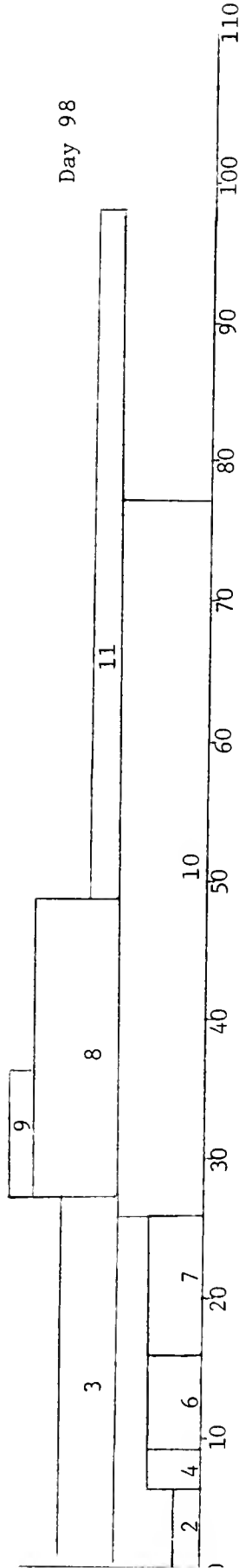
jobs. The routine uses a list of unscheduled jobs, whose predecessors have all been scheduled, ordered by Late Start and an ordered list of finish times of scheduled jobs. Thus, the routine steps through time to only those days when a change of resource usage is possible. On such a day it examines the list of available jobs, and schedules them in order of increasing Late Start. When the resources are exhausted the routine moves to the day of the next job completion and continues the scheduling process. As implied at the beginning of this section, examples can be constructed to favor or penalize either approach.

In figure 3 the heuristic of Moder and Philips is illustrated, first using an Early Start ordered list, then a Late Start ordered list as suggested by the authors. An examination of the resulting schedules shows that for this problem the Late Start heuristic is superior.

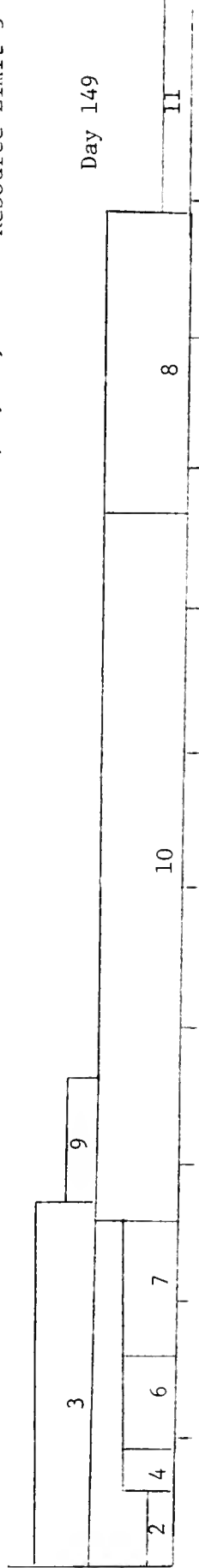


# RESOURCE CONSTRAINED NETWORKS

(1) SCHEDULE ALL JOBS AT E.S.



(2) SCHEDULE JOBS IN ORDER OF E.S. - 1, 2, 3, 4, 5, 6, 7, 10, 8, 9, 11, 12 Resource Limit 5



(3) SCHEDULE JOBS IN ORDER OF L.S. - 1, 3, 2, 4, 8, 6, 5, 7, 10, 11, 12 Resource Limit 5

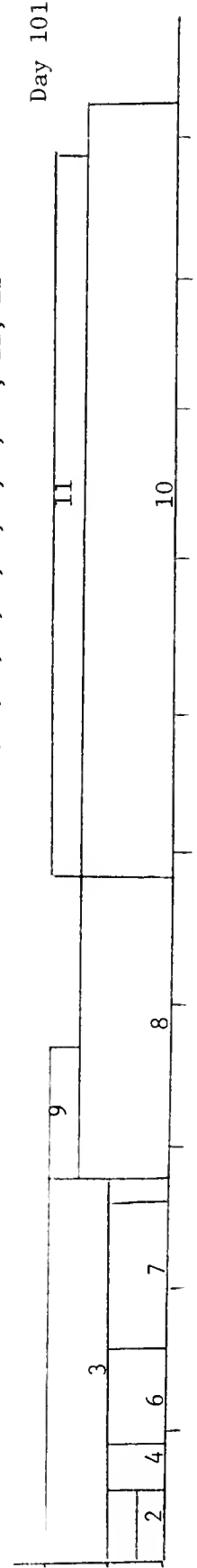


FIGURE 3





### Models with Time and Interdependency Constraints

The models we will discuss in this category are usually termed "time-cost" trade-off models. Some of the tasks in the project may be performed in several ways, such that the cost of a job increases with decreasing job time. It is usual to impose a due date, DD, and impose a penalty (\$p per day) for exceeding the due date, and it is also possible to include a reward (\$r per day) for completion before the due date. The problem then is to find the optimal way to perform each job so as to minimize the total of job and completion costs. Kelley and Walker [9] make the assumption that there is a linear relation between time,  $t_i$ , and cost,  $c_i$ , for task  $S_i$  with lower and upper bound on  $t_i$  of  $L_i$  and  $U_i$  as illustrated in Figure 4.

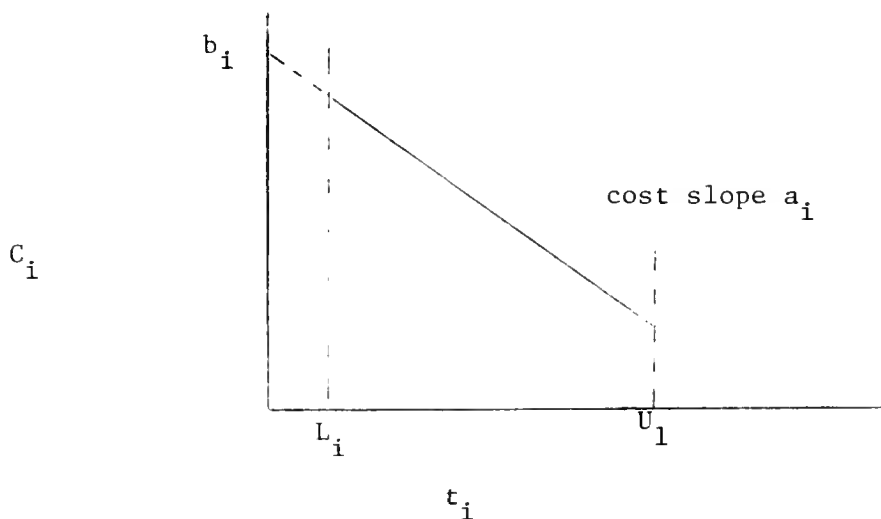


FIGURE 4



Thus, if there are N jobs in the project, we have the following general linear programming statement of the problem:

$$\text{Minimize } \sum_{i=1}^N (b_i - a_i t_i) - r(w_F^-) + p(w_F^+)$$

subject to the following general types of constraints.

Precedence

$$w_i + t_i \leq w_m$$

Task Length

$$t_i \leq U_i$$

$$t_i \geq L_i$$

Due Date

$$w_F - w_F^+ + w_F^- = DD$$

$$t_i, w_i, w_F^+, w_F^- \geq 0$$

where  $a_i$  = the slope of the cost curve

$w_F$  = the early start of  $S_F$ , an artificial finish job

$w_F^+$  = the number of days by which the project due date is exceeded

$w_F^-$  = the number of days before the due date that the project is completed

Again, linear programming is not the preferred method of solving this problem, but rather, a network flow algorithm has been derived for this purpose [8].



In some instances it may be unrealistic to observe a continuous relationship between time and cost for a task in a project. For example, if a job may only be performed by an eight-man crew on regular time, or by an eight-man crew on regular time plus two hours overtime, there are two discrete ways to perform the job (two mutually exclusive alternatives). The problem of discrete alternatives in time-cost trade-off problems is called Decision CPM and is described in references [2, 3, 4].

An integer programming formulation of this problem for the network of Figure 5 is now given.

$$\begin{aligned} \text{Minimize} \quad & 180d_{7,1} + 100d_{7,2} + 260d_{8,1} \\ & + 150d_{8,2} + 150w_F^+ - 25w_F^- + k \end{aligned}$$

Subject to the following constraints

Precedence

$$\begin{aligned} w_1 + t_1 &\leq w_3 \\ -M(1-d_{8,1}) + w_{8,1} + t_{8,1} &\leq w_9 \quad \text{etc.} \end{aligned}$$

(where M is a large positive number).

Interdependency

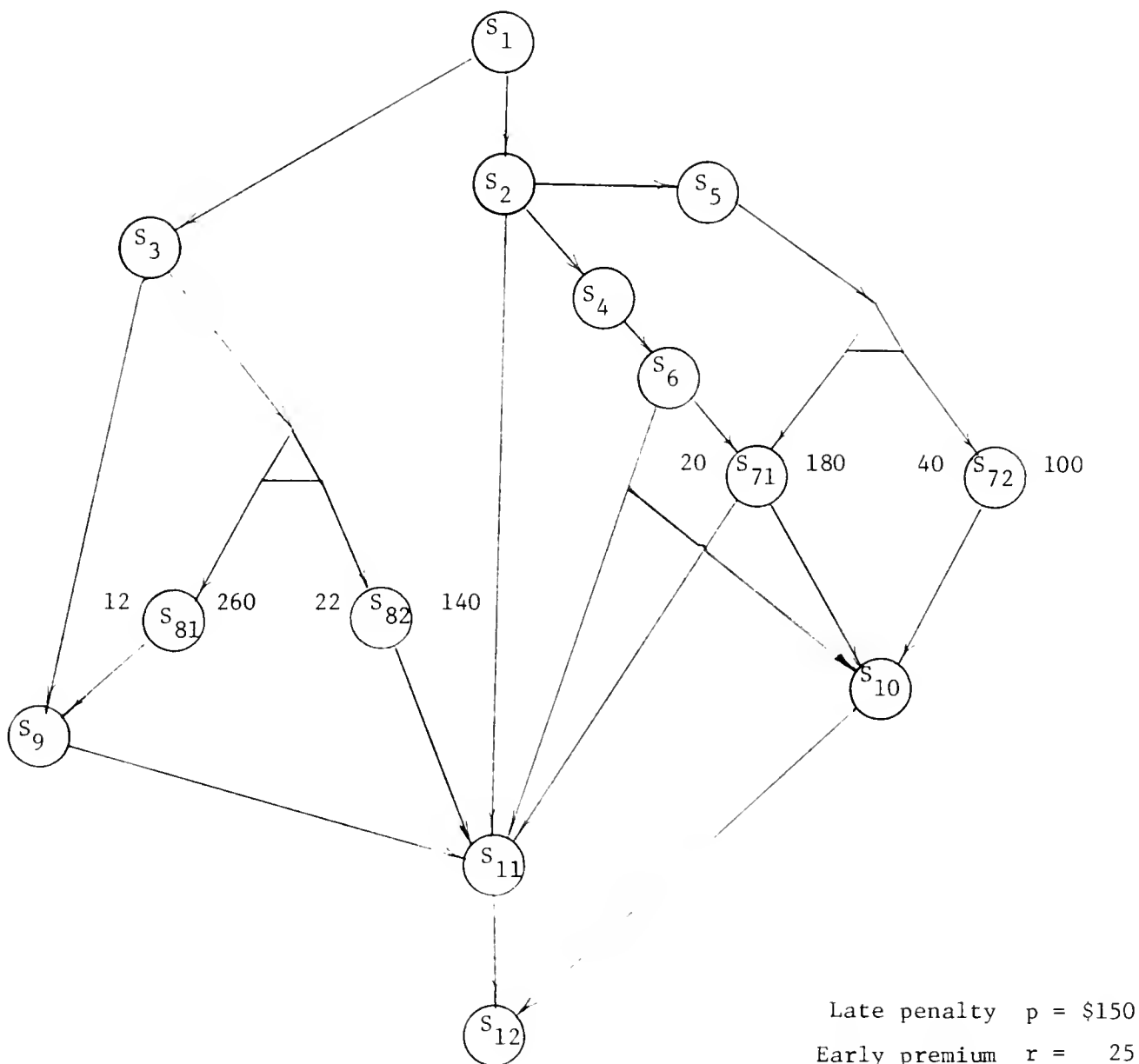
$$\begin{aligned} d_{7,1} + d_{7,2} &= 1 \\ d_{8,1} + d_{8,2} &= 1 \end{aligned}$$

and

$$d_{8,1} \geq d_{7,2}$$



DECISION C.P.M.



Late penalty  $p = \$150$

Early premium  $r = 25$

INTERDEPENDENCY

$S_{7,1}$  or  $S_{7,2}$

$S_{8,1}$  or  $S_{8,2}$

$S_{7,2}$  only if  $S_{8,1}$

FIGURE 5





Due Date

$$w_{12} - w_{12}^{+} + w_{12}^{-} = DD$$

$$w_i \geq 0 \quad d_{ij} \text{ integer } (0,1)$$

where the integer variables

$$d_{ij} = \begin{cases} 1 & \text{if task } S_{ij} \text{ is performed} \\ 0 & \text{otherwise} \end{cases}$$

Note that the effect of  $-M(1 - d_{8,1})$  is to effectively remove the precedence constraint between  $S_{8,1}$  and  $S_9$  if  $S_{8,1}$  is not performed.

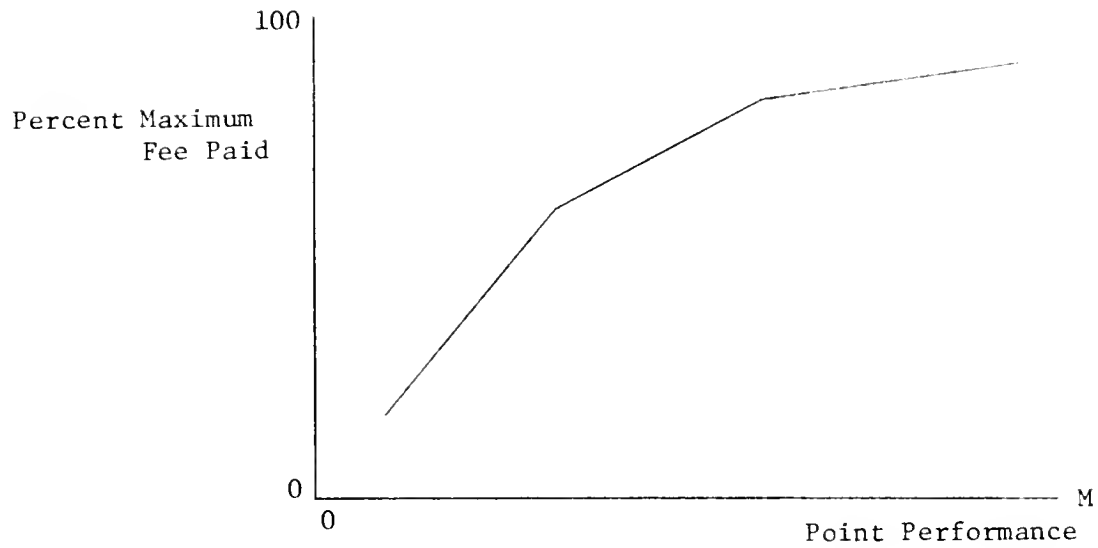
In the past several years, some government agencies have changed their contracting procedure from predominantly cost plus fixed fee to incentive fee contracts. The contracts are written so that the maximum fee obtainable increases as performance on the contract increases. Performance is measured by the number of points awarded for successfully passing quality tests, meeting specified due dates within the project and achieving required operating characteristics. The percent of the maximum fee that is actually paid decreases with increasing cost of the project.

A sample cost structure is shown in Figure 6.

A manufacturer faced with a time-cost trade-off problem within an incentive contract has an especially difficult problem. If a task is "crashed" or performed in less than the usual time, it is possible that extra points would be earned as a result of meeting a particular due date, but at the same time, the increase in project cost will cause all points to be devalued.



RELATION BETWEEN POINT PERFORMANCE  
AND PERCENT MAXIMUM FEE OBTAINED



RELATION BETWEEN MAXIMUM OBTAINABLE FEE  
AND COST PERFORMANCE

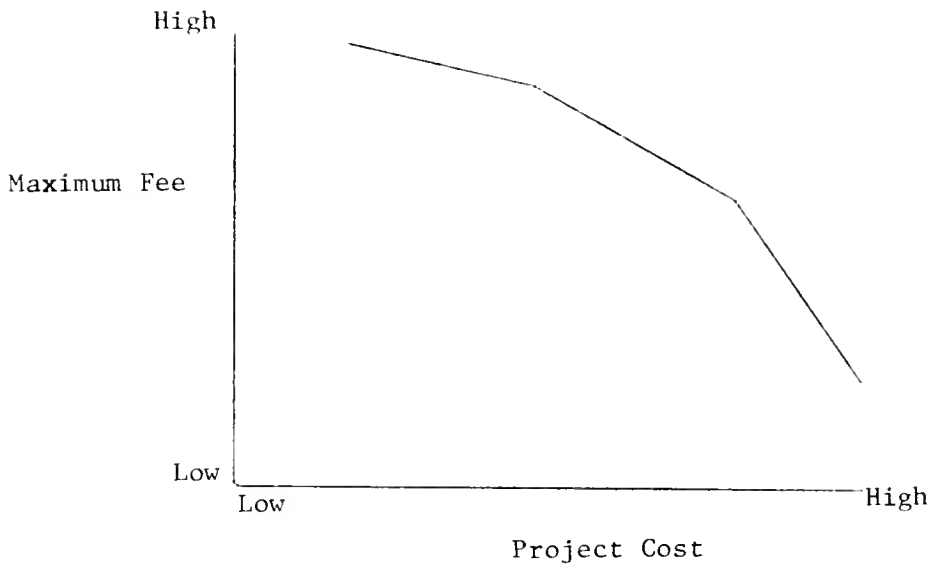


FIGURE 6



The following integer linear programming formulation [2], which is an extension of the Decision CPM model, was applied to a NASA project by R. Smylie [14].

Maximize

$$\sum_{i \in S_m} r_i w_i^- - p_i w_i^+$$

Subject to the following constraints.

Time as above

Interdependence as above

Due Date

$$w_i^- - w_i^+ + w_i^- = DD_i \quad i \in S_m$$

$$\text{Budget} \quad \sum_{i=1}^n \sum_{j=1}^{k(i)} c_{ij} d_{ij} \leq B$$

where

$S_m$  = the set of all milestones that are rewarded ( $r_i$  points)  
or penalized ( $p_i$  points)

$k(i)$  = the number of discrete alternatives for job  $S_i$

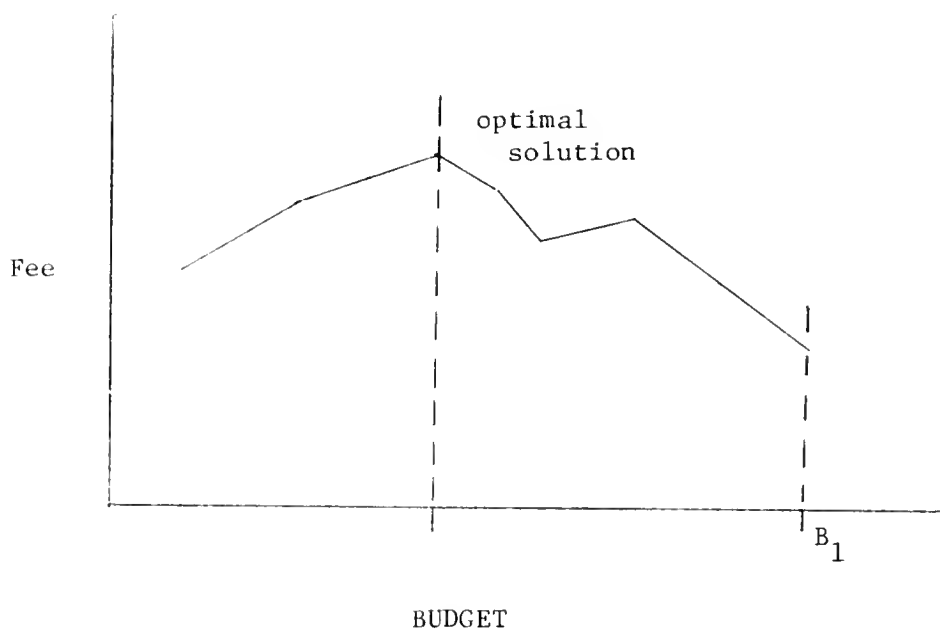
$B$  = a budget limit on the project.

It is interesting to note in this formulation that the budget constraint  $B$  applies not by period, but over the life of the project, in much the same way as an aircraft weight constraint applies across the complete airplane.



The formulation maximizes the number of performance points obtained subject to cost limits on the whole project. Initially the problem is solved without a budget constraint to determine the maximum available points and the cost associated with this solution. B is then set one unit below the cost found above, and the problem is resolved. The routine is applied iteratively until the minimum cost, minimum point solution is reached.

For each solution the combination of performance points obtained and budget cost will allow a total fee to be calculated. Figure 7 shows a series of such points with the optimal solution marked.







Although we have formulated these time-cost trade-off models as integer programming problems, current algorithms will not allow them to be solved efficiently in that way. Several tree-search algorithms [4] have been developed and tested for problems of this type. Each routine begins with a "reduced" decision network which is simply the original network with all non-decision jobs eliminated and replaced with maximal timespan between each pair of decision jobs and between decision jobs and the start and finish nodes of the network. In this reduced network the Early Start time of all decision jobs will be the same as the Early Start times of jobs in the original network, and an optimal solution to one is an optimal solution to the other. Figure 8 shows the decision network of Figure 5 in reduced form. The tree search method, or branch and bound technique, essentially enumerates all potential solutions which are proven to be sub-optimal or infeasible.

The method is applied to our sample problem in Figure 9, with a due date of day 97. Before any decisions are made, it is obvious from the reduced network that the minimum total job cost for any solution would be  $C_{7,2} + C_{8,2} = \$240$ . In the first level of the tree we assume the choice of  $S_{8,1}$  and  $S_{8,2}$ . The choice of  $S_{8,1}$  gives a minimum path length of 96 days and a minimum job cost of  $C_{7,2} + C_{8,1}$ . The overall minimum and for this solution is

$$C_{7,2} + C_{8,1} + (w_{12} - DD)(r) = 100 + 260 + (96 - 97)25 = \$335.$$



REDUCED NETWORK

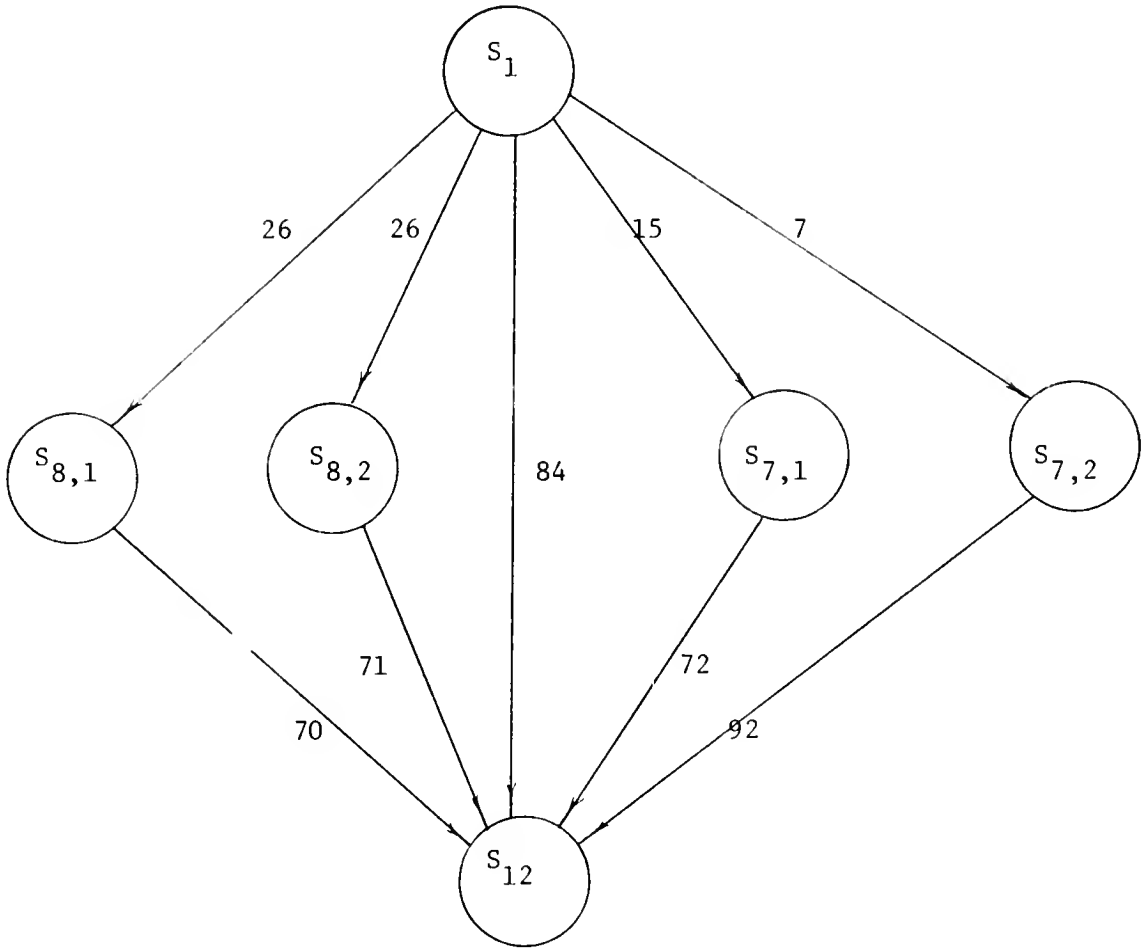


FIGURE 8



Similarly for the choice of  $S_{8,2}$  our absolute minimum is

$$140 + 100 + (97-97)25 = \$240.$$

At this point the solution with  $S_{8,2}$  seems to be preferable and so the solution is extended. The combination  $S_{8,2}$  and  $S_{7,2}$  is infeasible given the original interdependence constraint, and therefore we need only evaluate  $S_{8,2}$  plus  $S_{7,1}$ . The cost of this solution is

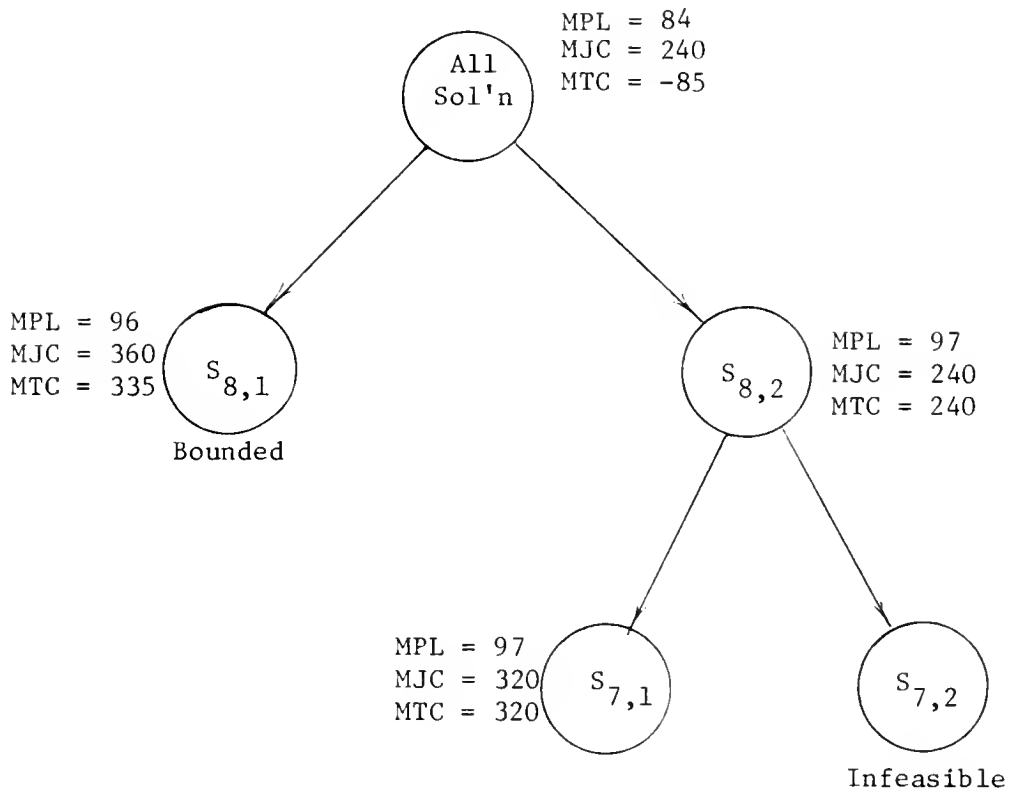
$$180 + 140 + (97-97)25 = \$320.$$

Since this complete solution is cheaper than the lower bound on all solutions that contain  $S_{8,1}$ , the optimal solution to the problem is the choice of  $S_{7,1}$  and  $S_{8,2}$ . Computational experience with algorithms of the type illustrated here shows that they are efficient for decision network problems. For problems with fifteen decision nodes, three alternatives at each node, times in the range of 5-50 seconds are usual. [4]

In many instances a large project may be simply a collection of relatively independent subprojects. Experience has shown that there is an exponential growth of solution time with increases in the number of decision nodes. Therefore it is faster to solve a number of small problems and combine these solutions in a master problem, rather than attempt to solve the large problem directly. This decomposition strategy is similar to the decomposition principle for large linear programs. We define a subproject here as a collection of decision nodes in a network that have a single starting point and a single finish.



# BRANCH AND BOUND METHOD



M.P.L. = Minimum Project Length

M.J.C. = Minimum Job Cost

M.T.C. = M.P.L. + M.J.C.

FIGURE 9

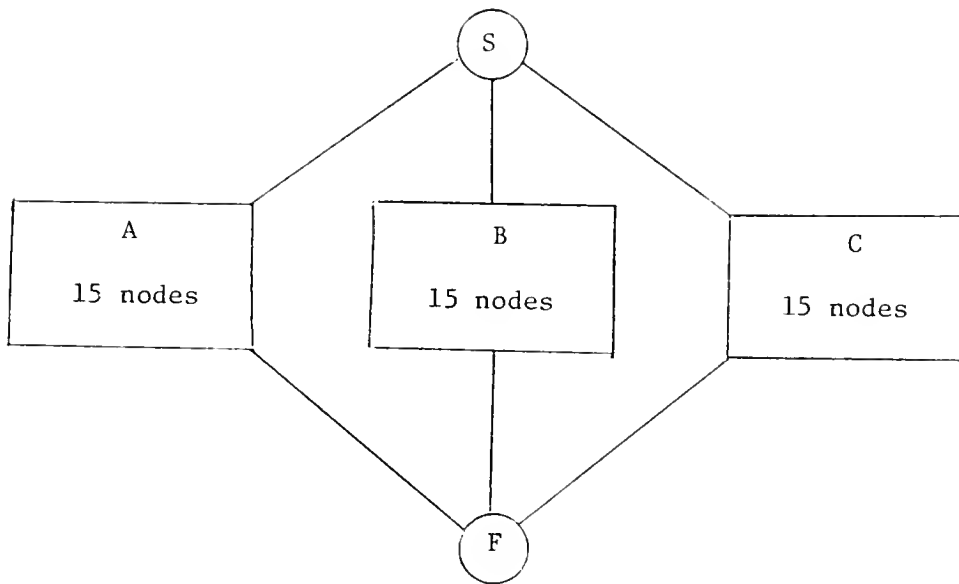




Due to the interaction between subprojects and the rest of the project, it is not possible to solve for a single solution in each subproject and then use these solutions to obtain an overall optimum. A subproject would not ordinarily have an independent criterion function. Instead, the criterion function is concerned with the whole project's completion date, and the effect of a subproject solution upon that date is not known until decisions are made for all other parts of the network. The approach used is based on the observation that the set of all feasible solutions to a subproject has all the characteristics of a decision node. That is, each undominated feasible solution has a time, a cost, a set of predecessors and a set of successors, and only one such solution may be accepted in the final solution to the complete project.

The procedure then is to solve each subnetwork for the set of all undominated solutions, then replace the subnetwork with a decision node in which each alternative is an undominated solution of the subnetwork. The project network then contains substantially fewer decision nodes and may be more easily solved. The example in Figure 10 illustrates the potential usefulness of this method. [4]





### Solution Times

#### With Decomposition

A	3.8 seconds
B	13.3 seconds
C	3.4 seconds
Master Problem	1.8 seconds
Total	<u>22.3 seconds</u>

#### Without Decomposition

A,B,C                      163 seconds

FIGURE 10



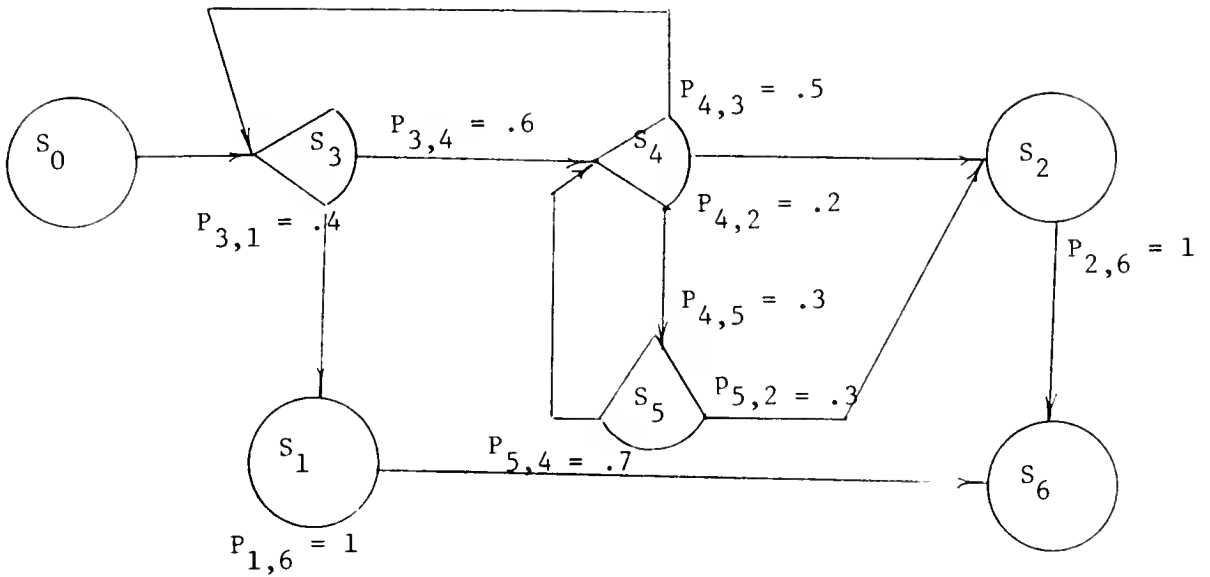
## Probabilistic Network Models

Many design tasks are not well-defined sequential processes, as we might expect in a repetitive production process. As a project develops from a general problem statement to a final set of blueprints, the design of any subsystem will be subjected to a series of tests for weight, stress, cost or reliability, as we discussed in our introduction. If a constraint is violated, then it may be necessary to redesign some aspect of the subsystem. Thus, the sequence of tasks is not at all certain but depends on probabilistic design and research outcomes. We will now introduce a graphical model for displaying these problems, then show how we may mathematically solve for the expected duration (critical path length) for such a process.

In Figure 11a we have a set of tasks  $S_0 - S_6$ , each with known performance times and known probabilities of being succeeded by other tasks in the set. These relations may be represented by the following transition matrix, which again shows the probability of going to every job from every job. Job  $S_6$  is omitted here since we know that both jobs  $S_1$  and  $S_2$  lead directly to it and that it will be performed exactly one time.



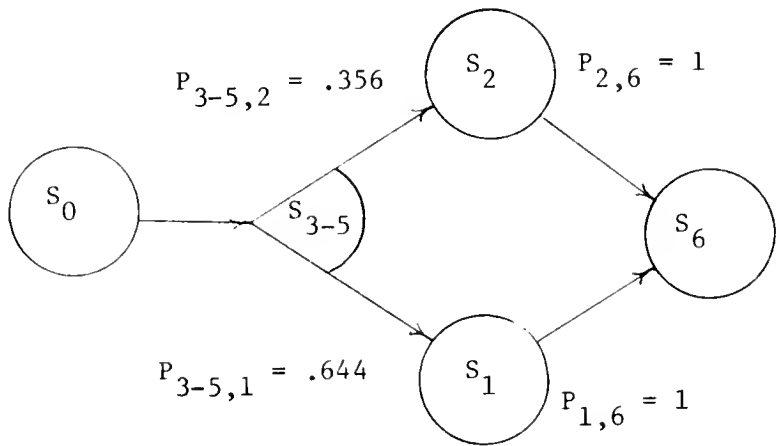
PROBABILISTIC NETWORKS



(a)

and  $t_0 = 0$  days

- $t_1 = 4$
- $t_2 = 6$
- $t_3 = 3$
- $t_4 = 1$
- $t_5 = 2$
- $t_6 = 0$



$$t_{3-5} = 6.79$$

(b)

FIGURE 11





		1	2	3	4	5
Transition Probability Matrix	1	1	0	0	0	0
	2	0	1	0	0	0
	3	.4	0	0	.6	0
	4	0	.2	.5	0	.3
	5	0	.3	0	.7	0

We now digress to state some results from the theory of Markov chains with absorbing states. Given the transition matrix  $P$  of an absorbing Markov Chain with  $m$  absorbing states  $S_1 - S_m$  and  $n$  non-absorbing states  $S_{m+1} - S_{m+n}$ , we can partition  $P$  as follows:

$$P = \begin{array}{c|ccc} & \begin{array}{c} S_1 \\ S_m \\ S_{m+1} \\ S_{m+n} \end{array} & \begin{array}{c} S_m \\ S_{m+1} \end{array} & \begin{array}{c} S_{m+1} \\ S_{m+n} \end{array} \\ \hline \begin{array}{c} S_1 \\ S_m \\ S_{m+1} \\ S_{m+n} \end{array} & \begin{array}{c} J \\ R \end{array} & \begin{array}{c} 0 \\ Q \end{array} \end{array}$$

where

$J$  is an  $m \times m$  identity matrix

$0$  is an  $m \times n$  null matrix (all zeros)

$R$  is an  $n \times m$  matrix

$Q$  is an  $n \times n$  matrix.



We define the fundamental matrix  $N$  as  $N = (I - Q)^{-1}$  where  $I$  is an  $n \times n$  identity matrix and the symbol  $( )^{-1}$  implies that we take the inverse of the matrix  $I - Q$ . The elements of the matrix  $N$ , that is,  $n_{ij}$ , will be the average number of times state  $S_j$  is visited, given that the process begins in state  $S_i$ . Furthermore, we may define the matrix

$$B = N \cdot R$$

where  $R$  is the  $n \times m$  partition of the transition matrix. The elements of  $B$ , that is,  $b_{ij}$ , give the probability that the system will end in absorbing state  $S_j$ , given that it begins in state  $S_i$ .

We will now compute and interpret the matrix  $N$  and  $B$  for our problem.

$$I - Q = \begin{matrix} & \begin{matrix} 3 & 4 & 5 \end{matrix} \\ \begin{matrix} 3 \\ 4 \\ 5 \end{matrix} & \begin{bmatrix} 1 & -.6 & 0 \\ -.5 & 1 & -.3 \\ 0 & -.7 & 1 \end{bmatrix} \end{matrix} \qquad N = \begin{matrix} & \begin{matrix} 3 & 4 & 5 \end{matrix} \\ \begin{matrix} 3 \\ 4 \\ 5 \end{matrix} & \begin{bmatrix} 1.61 & 1.225 & .367 \\ 1.02 & 2.04 & .612 \\ .714 & 1.43 & 1.43 \end{bmatrix} \end{matrix}$$

$$B = N \cdot R$$

$$= \begin{bmatrix} 1.61 & 1.225 & .367 \\ 1.02 & 2.04 & .612 \\ .714 & 1.43 & 1.43 \end{bmatrix} \qquad \begin{bmatrix} .4 & 0 \\ 0 & .2 \\ 0 & .3 \end{bmatrix}$$

$$= \begin{matrix} & \begin{matrix} 1 & 2 \end{matrix} \\ \begin{matrix} 3 \\ 4 \\ 5 \end{matrix} & \begin{bmatrix} .664 & .356 \\ .408 & .592 \\ .285 & .715 \end{bmatrix} \end{matrix}$$



Thus, if we begin the process at task  $S_0$ , we will perform

$S_3$  1.61 times an average,

$S_4$  1.225 times, and

$S_5$  .367 times.

Given the task time spent on each of these activities, we would expect to spend

$$(1.61)(3) + (1.225)(1) + (.367)(2) = 6.79 \text{ days}$$

on tasks  $S_3$ ,  $S_4$  and  $S_5$ . Note that the network can be redrawn as in Figure 11b. The probabilities  $P_{3-5,1}$ ,  $P_{3-5,2}$  are taken directly from the B matrix. Note that this Markov decomposition is quite parallel to the decomposition of decision networks that we discussed earlier. It is also interesting to note that the concepts of reduction apply here as well. Task  $S_4$  could possibly represent a complex deterministic network with critical path length  $t_4$ . One point of caution is necessary. The time we have calculated for  $t_{3-5}$  in this problem is an average time to exit, independent of whether the exit is to  $S_1$  or  $S_2$ . For some decomposition problems it will be useful to calculate an expected time based on the task to which we exit [15].

### Implications for the Design of Information and Control Systems

In the discussion to this point we have centered on the possibility that the planning models allow some centralization of decision-making in project organizations. For example, the slack in the project could be distributed among tasks on the critical path so that each sub-group has



a clear time constraint on the finish of their activity. While the project is being planned, staffing decision could be made so as to balance the cost of extra people against the cost of the project running over the due date. Similarly, in the control of the project, if a sub-group is about to violate a time constraint, the models might be used to evaluate the effect of this slippage, and if necessary, find the cheapest way to recover the lost time. As has been suggested, the structure the models provide allows a more efficient processing of information on relevant dimensions to the management hierarchy.

Even if the models are not used formally in a decision system, they may be conceptually useful in the design of information collection procedures to support management decisions. Specifically, it is possible to infer from the models the level of aggregation of data that will be useful for certain decisions, and also they allow an estimate to be made of the value of information and the allowable period between the collection of updated information. Finally, the models give important insight into methods of decomposing problems into either independent sub-problems, or into sub-problems whose interdependence with the complete problem is well-defined.

In the solution procedure for Decision C.P.M. problems, the first step is to eliminate all non-decision jobs and replace them by the maximal distance between decision jobs. For our purpose, that is for planning or control decisions in the time-cost dimensions, the reduced network is exactly equivalent to the original network. Not only is the reduced network a more efficient base for our branch and bound algorithms, but it would





also be more comprehensible for a manager making trade-off decisions. The essence of this point is that for many decisions it is enough to report that summary piece of information, the length of the critical path.

We will now consider the determination of the value of information and the question of optimal periods between collection of the information. As we have stated, the need for a time-cost trade-off decision would be signalled by the violation of a time constraint by some sub-group in the project. There is obvious advantage to receiving this information early in that the number of alternatives available to ease the problem will be greater. This implies that for tasks on or near the critical path we should have frequent information on their progress. For areas of the network with substantial slack it may only be necessary to report major milestones. The design of an optimal control scheme would be a difficult task, but at least our knowledge of simple project models gives insight into the general characteristics of such a scheme. It should be noted that the same type of comment might be made about tasks that use critical resources and those that don't.

The concept of decomposition, the factoring of large problems into a series of small problems, also has implications for the design of information systems. In an actual decision network [14], it was found that relatively few of the sub-projects could actually influence the length of the critical path. The others had so much slack that it was possible to treat them as independent projects with no due date penalty. This would allow the complete decentralization of decision-making to the organization unit responsible for the sub-unit. This implies that information for control of the sub-project



could be collected and reported locally.

We have confined this discussion to the dimensions that are of primary interest in this paper, time and cost. However, to return to the introductory comments, if models can be made of trade-offs in other dimensions, then the concepts of aggregation, value of information and decomposition will be useful in introducing those models into a control system. For example, in a weight-cost trade-off model a sub-group might only have to report the most favorable marginal trade-off available to them. In a reliability-cost trade-off model we might find that a sub-system is completely independent. Until such systems are available, management will have to use their judgement to combine the quantifiable dimensions of time and cost with less structured dimensions of their problems.



BIBLIOGRAPHY

- [1] Balinski, M. L., "Integer Programming: Methods, Uses, Computation," Management Science, Vol. 12, No. 3, November 1965.
- [2] Crowston, W. B., "Decision Network Planning Models," Management Sciences Research Report No. 138, Carnegie-Mellon University, May 1968.
- [3] Crowston, W. B. and G. L. Thompson, "Decision CPM: A Method for Simultaneous Planning, Scheduling and Control of Projects," Operations Research, Vol. 15, No. 3, May-June 1967.
- [4] Crowston, W. B. and M. Wagner, "A Comparison of Tree Search Schemes for Decision Networks," Working Paper No. 380-69, S.S.M., M.I.T., April 1969.
- [5] Davis, E. W., "Resource Allocation in Project Network Models - A Survey," The Journal of Industrial Engineering, Vol. XVII, No. 4, April 1966.
- [6] "D.O.D. and N.A.S.A. Guide PERT/COST, Office of the Secretary of Defense and National Aeronautics and Space Administration, Washington, D.C., 1962.
- [7] Elmaghraby, S. G., "An Algebra for Analysis of Generalized Activity Networks," Management Science, Vol. 10, No. 3, April 1964.
- [8] Ford, L. P. Jr. and D. R. Fulkerson, Flows in Networks, Princeton University Press, 1963.
- [9] Kelley, J. E. Jr. and M. R. Walker, "Critical Path Planning and Scheduling," 1959 Proceedings of the Eastern Joint Computer Conference.
- [10] Kemeny, J. G. H. Mirkil, J. L. Smell and G. L. Thompson, Finite Mathematical Structures, Prentice-Hall Inc., 1959.
- [11] Levy, F. K., G. L. Thompson and J. D. Wiest, "Mathematical Basis of the Critical Path Method," Industrial Scheduling, Thompson and Muth, Eds., Prentice-Hall Inc., 1963.
- [12] \_\_\_\_\_, "Multi-Ship, Multi-Shop Workload Smoothing Program," Naval Research Logistics Quarterly, Vol. 9, No. 1, March 1962.
- [13] Pritsker, A. B., and W. W. Happ, "GERT: Graphical Evaluation and Review Technique, Part I. Fundamentals," The Journal of Industrial Engineering, Vol. XVII, No. 5, May 1966.



Bibliography, Cont'd.

- [14] Smylie, R. E., "The Application of Decision CPM to Incentive Contracts," .M. Thesis, S.S.M., M.I.T., 1967.
- [15] Thompson, B. U., "Analysis of Ship Design Processes by Extended Markov Chain Techniques," S.M. Thesis, M.I.T., 1969.
- [16] Wiest, J. P., "A Heuristic Model for Scheduling Large Projects with Limited Resources," Ph.D. Thesis, Carnegie Institute of Technology, 1962.
- [17] Kelley, J. E. Jr., "The Critical Path Method: Resources Planning and Scheduling," Industrial Scheduling, Muth and Thompson, Eds., Prentice-Hall Inc., 1963.
- [18] Moder, J. J. and C. R. Philips, Project Management with CPM and PERT, Rheinhold Publishing Company, 1964.

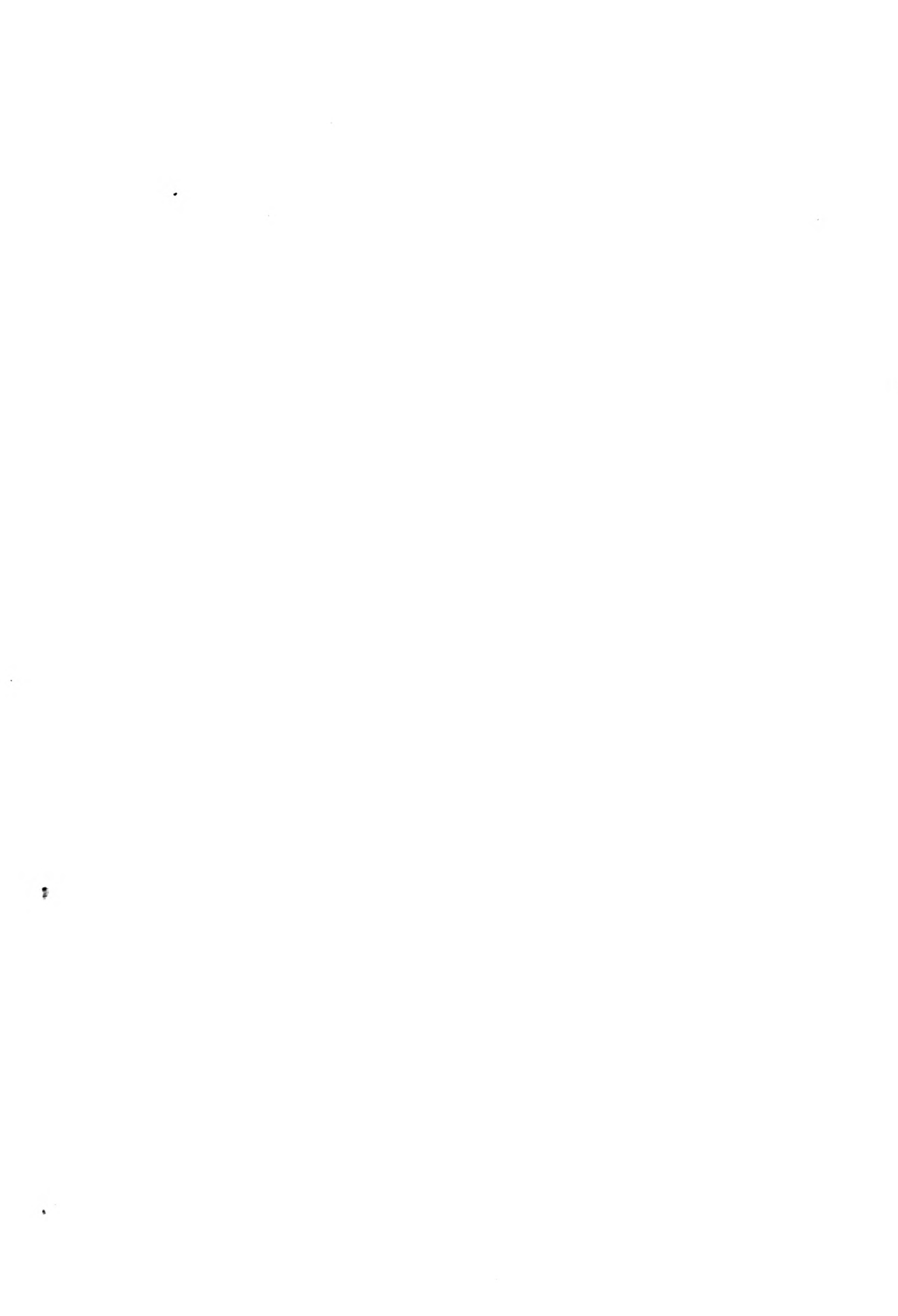
10-17

10-17









# SECRET Date Due

<del>MAY 3 '76</del>	AUG 29 '83	
<del>FEB 14 '76</del>	<del>MAY 2 '83</del>	
DEC 07 '78		
MAR 3 '77		
MAR 28 '77		
APR 22 '77		
MAY 11 '79		
APR 3 '79		

Lib-26-67

MIT LIBRARIES



45-70

3 9080 003 875 660

MIT LIBRARIES



454-70

3 9080 003 906 663

MIT LIBRARIES



456-70

3 9080 003 906 630

MIT LIBRARIES



457-70

3 9080 003 906 606

MIT LIBRARIES



458-70

3 9080 003 906 671

MIT LIBRARIES



459-70

3 9080 003 875 652

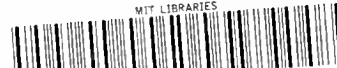
MIT LIBRARIES



460-70

3 9080 003 906 622

MIT LIBRARIES



461-70

3 9080 003 875 611

MIT LIBRARIES



462-70

3 9080 003 875 637

MIT LIBRARIES



463-70

3 9080 003 875 678

MIT LIBRARIES



464-70

3 9080 003 875 645

MIT LIBRARIES



465-70

3 9080 003 906 580

HD2

.M41

Nos. 4

Nos. 46

